86805: Software Architectures for Robotics
Localization system for a wheeled humanoid robot

Date: Wed 23:55, 01.02.16

Prof. F. Mastrogiovanni, PhD C. Recchiuto

Rabbia Asghar, BEng, Ernest Skrzypczyk, BSc

Contents

Rollo - Humanoid robot

Odometry

Odometry is currently the most widely used technique for determining the position of a mobile robot. It mainly involves use of various encoders, for example on wheels, as sensors to estimate the robot's position relative to a starting or previous location. Usually, it is used for real-time positioning in the between the periodic absolute position measurements, for example GPS (Global Positioning System) provides absolute position feedback, however it updates at $0.1 \div 1s$ interval, during which odometry could be used for localization. One of the major downsides of odometry is its sensitivity to errors. There are various error sources discussed in section Odometry errors on page 4 in detail. One significant source of error influencing the accuracy of odometry that is worth mentioning however, is the integration of velocity measurements over time to give position estimates.

First the odometry based motion model for the robot is derived. The model is derived based on the following important assumptions:

1. The robot is a rigid body
2. The model represents a differential drive robot
3. The wheels of the robot are perfect discs and have the same diameters
4. the wheels have no thickness
5. There is no slip in the wheels
6. Both wheels are turning in the forward direction

A differential drive robot runs straight when the linear speed of both the left and right wheel is same. If the speed of one wheel is greater than the other, the robot runs in an arc. This derivation can be divided in three distinct cases of robot motion:

The basic premise for the odometry model of the Rollo humanoid robot is presented in figures (**??**) and (**??**).

1. Clockwise direction
2. Counterclockwise direction
3. Straight line

Clockwise direction

First, the case is considered when the speed of left wheel is greater than the right, and the robot will run in clockwise direction. Both right and left wheel will rotate around the same center of a circle.

P_i represents the initial position of the robot, defined by the center of the line joining two wheels, while l represents wheel base, ergo distance between two wheels. S_L and S_R represent the distance travelled by left and right wheel respectively. $\Delta\Theta$ represents the angle of travel for both the wheels and r is the radius of travel from the center of robot. With encoder feedback from left and right wheel, S_L and S_R can simply be computed using equations (1) and (2)

$$S_L = \frac{n_L}{60} 2\pi r_L \tag{1}$$

$$S_R = \frac{n_R}{60} 2\pi r_R \tag{2}$$

where n_L and n_R are the revolutions per minute ([rpm]) of left and right wheel, and r_L and r_R are the radii of the 2 wheels.

S_L and S_R can be related to r and $\Delta\Theta$ using formulas (3) and (4)

$$S_L = (r + \frac{l}{2})\Delta\Theta \tag{3}$$

$$S_R = (r - \frac{l}{2})\Delta\Theta \tag{4}$$

The equations (3) and (4) on page ??an be solved simultaneously to compute $\Delta\Theta$.

$$\Delta\Theta = \frac{S_L - S_R}{l} \tag{5}$$

The length between initial position, P_i and final position, P_f can be approximated by taking average of S_L and S_R.

$$\Delta S = \frac{S_L + S_R}{2} \tag{6}$$

Final position of the robot and its orientation can then be derived using basic trigonometry and geometry relations as displayed in equation (7):

$$P_f = \left[P_{ix} + \Delta S(\cos(\Theta_i - \tfrac{\Delta\Theta}{2})), \quad P_{iy} + \Delta S(\sin(\Theta_i - \tfrac{\Delta\Theta}{2})\right] \tag{7}$$

$$\Theta_\mathrm{f} = \Theta_\mathrm{i} - \Delta\Theta$$

where Θ_f and Θ_i are the final and initial orientation respectively.

Counterclockwise direction

Similar derivation of the robot can be derived for counterclockwise rotation, when the rpm of right wheel is higher than that of the left wheel.

$$\Delta\Theta = \frac{S_R - S_L}{l} \tag{8}$$

The relations for final position and rotation are as follows for this case:

$$P_f = \left[P_{ix} + \Delta S(\cos(\Theta_i + \tfrac{\Delta\Theta}{2})), \quad P_{iy} + \Delta S(\sin(\Theta_i + \tfrac{\Delta\Theta}{2})\right] \tag{9}$$

$$\Theta_\mathrm{f} = \Theta_\mathrm{i} + \Delta\Theta$$

where Θ_f and Θ_i are the final and initial orientation respectively.

Straight line motion

In case of straight line motion, the orientation of the robot stays the same. Thus, Θ is this case is 0. Also, S_L is the same as S_R.

$$P_f = \left[P_{ix} + S_L(\cos(\Theta_i)), \quad P_{iy} + S_L\sin(\Theta_i)\right] \tag{10}$$

$$\Theta_\mathrm{f} = \Theta_\mathrm{i}$$

Final set of equations

With the derivation of formulas understood for all three cases,for the ease of computation it is desirable to have the same set of equations in all cases.

The equations selected for this model are selected as follows.

$$\Delta\Theta = \frac{S_L - S_R}{l} \tag{11}$$

$$\Delta S = \frac{S_L + S_R}{2} \tag{12}$$

$$P_f = \left[P_{ix} + \Delta S(\cos(\Theta_i - \tfrac{\Delta\Theta}{2})), \quad P_{iy} + \Delta S(\sin(\Theta_i - \tfrac{\Delta\Theta}{2})\right] \tag{13}$$

$$\Theta_\mathrm{f} = \Theta_\mathrm{i} - \Delta\Theta$$

Adaption of Odometry Model for Rollo

Currently, the encoders feedback is unavailable in the robot and true odometry model can not be implemented. However, an attempt has been made to implement its modified version. The distance covered by right and left wheels is instead estimated from the control command.

Straight line motion

Because of the mechanical difference in the two legs, it is almost impossible for the two wheels to run at same speed . Moreover, the two motors/wheels are powered using 2 different LIPO packs and are run in open loop. The difference in voltage level further results in different speed of the motor at the same pwm. This is a major source for deviation in behavior of robot's adjusted odometry model.

Rollo was run in a straight line at different commands and its location feedback from the motion capture system was logged. For a certain command, average speed of left and right wheel was determined from the logs.

Odometry errors

As briefly discussed above, odometry is very sensitive to errors. Results of an odometry based models can be improved by rapid and accurate data collection, equipment calibration, and processing.

The sources of odometry errors can be broadly divided into two categories.

1. Systematic errors
2. Non-systematic errors

Systematic errors

Systematic errors are caused by inherent properties of the system. They are usually caused by imperfections in the design and mechanical implementation of a mobile robot. On most smooth indoor surfaces, systematic errors play a much bigger role in odometry errors than non-systematic errors.Also, systematic errors accumulate constantly.

The major sources of the systematic error in an odometry model of a robot are unequal wheel diameters and the uncertainty about the effective wheelbase.

Rollo consists of 4 rubber tires, 2 on each side. Rubber tires help improve traction but they are difficult to manufacture to exactly the same diameter. Also, with time, rubber tires compress differently under asymmetric load distribution. The diameters are used in computing S_L and S_R as previously shown in equations 1 and 2.

Furthermore, the contact area between the tires of Rollo and the floor is very wide. This gives trouble in determining the wheelbase of the robot. Wheel base is used in computation of angle of travel as previously shown in equations 5.

Other Examples of sources of these errors include

1. Average of both wheel diameters differ from nominal diameter
2. Misalignment of wheels
3. Limited resolution during integration (encoder sampling rate, encoder measurement resolution)

Non-systematic errors

Non-systematic errors are caused by unknown and unpredictable changes in the system or in the environmental conditions. In mobile robots, they are usually caused by uneven floors. On rough surfaces with significant irregularities, non-systematic errors may be more dominant. On contrary to systematic errors, non-systematic errors do not accumulate with time.

Some examples of non-systematic errors are as follows:

1. Travel over uneven floors or unexpected objects on the floor
2. Wheel-slippage due to:
   (a) slippery floors
   (b) over-acceleration
   (c) fast turning (skidding)
   (d) non-point wheel contact with the floor

Measuring Odometry Errors

University of Michigan Benchmark test (UMBmark) procedure was used to measure odometry errors and calibrate the system.This procedure was developed by Johann Borenstein and Liqiang Feng around 1995 in the University of Michigan's Advanced Technologies Lab. Details of this test can be found in the research paper.

As discussed in the previous section, there are 2 dominant sources of systematic errors in odometry: unequal wheel diameters and the uncertainty about the effective wheelbase.

Unfortunately, we are estimating S_L and S_R from the command given to the robot due to unavailability of encoder feedback. That means equations 1 and 2 are not used in the modified model. Even if we are able to determine the error in the relative size of the wheels, it can not be corrected or calibrated in the current model.

However, error due to the uncertainty in effective wheelbase can be determined and calibrated in the model.An incorrect wheelbase value will result in incorrect computation of angle of travel, $\Delta\theta$ refer to 11. This error, E_b is directly proportional to the effective wheelbase divided by the nominal wheelbase, l :

$$E_b = \frac{effectivewheelbase}{l} \tag{14}$$

After computing this error, we can replace the nominal wheelbase in equation 11 by effective wheelbase. so we can correct for this error by multiplying the nominal wheelbase (bn) by the error factor to give us the effective wheelbase (be):

$$\Delta\Theta = \frac{S_L - S_R}{E_b * l} \tag{15}$$

E_b is determined using UMBmark procedure. The summary of the UMBmark setup for Rollo is as follows:

1. At the beginning of the run, measure the absolute position and orientation of the robot using motion capture system and initialize to that position the starting point of the vehicle's odometry program.
2. Run the vehicle through a 2×2 m square path in cw direction, making sure to
   (a) stop after each 2 m straight leg;
   (b) make a total of four 90 -turns on the spot;
   (c) run the vehicle slowly to avoid slippage.
3. Upon return to the starting area, measure the absolute position and orientation of the robot using motion capture system
4. Compare the absolute position to the robot's calculated position using equations 16, 17 and 18.

$$\epsilon x = x_a bs - x_c alc \tag{16}$$

$$\epsilon y = y_a bs - y_c alc \tag{17}$$

$$\epsilon\theta = \theta_a bs - \theta_c alc \tag{18}$$

5. Repeat steps 1-4 for four more times (i.e., a total of five runs).
6. Repeat steps 1-5 in ccw direction.
7. Compute coordinates of center of gravity for each cluster in cw and ccw direction using equations 19 and 20 where n is 5.

$$x_{c.g.,cw/ccw} = \frac{1}{n}\sum_{i=1}^{n} \epsilon x_{i,cw/ccw} \tag{19}$$

$$y_{c.g.,cw/ccw} = \frac{1}{n}\sum_{i=1}^{n} \epsilon y_{i,cw/ccw} \tag{20}$$

8. Compute E_b using equations 21,22 and 23

$$E_b = \frac{90^\circ}{90^\circ - \alpha} \tag{21}$$

where α can be computed as

$$\alpha = \frac{x_{c.g.,cw} + x_{c.g.,ccw}}{-4L}\frac{180^\circ}{\pi} \tag{22}$$

or

$$\alpha = \frac{y_{c.g.,cw} - y_{c.g.,ccw}}{-4L} \frac{180°}{\pi} \tag{23}$$

and L represents the side of square pathi.i.e. 2m.

System and Measurements Model

System Model

In order to implement Kalman Filter, the previously described model must be first represented in state space representation.

We can define the location of the robot at instant k using state variables. This will include position in x and y coordinates and orientation.

$$x_k = \begin{bmatrix} x_{[x],k} \\ x_{[y],k} \\ x_{[\theta],k} \end{bmatrix} \tag{24}$$

The control input provided to robot is the speed of right and left wheel. This defines the distance covered by both the wheels in unit time. The relative displacement of the robot at instant k can be notated by d_k. Using equations 9,7 and 10, relative displacement can be expressed in terms of r and θ. Thus, control input u_k can be expressed as a function of relative displacement.

$$u_k = j(d_k) \tag{25}$$

$$u_k = \begin{bmatrix} u_{[\Delta S],k} \\ u_{[\Delta\theta],k} \end{bmatrix}$$

Given x_{k-1} and u_{k-1}, the next location of the robot, x_k can be computed.

$$x_k = f(x_{k-1}, u_{k-1}) = \begin{bmatrix} f_x(x_{k-1}, u_{k-1}) \\ f_y(x_{k-1}, u_{k-1}) \\ f_\theta(x_{k-1}, u_{k-1}) \end{bmatrix} \tag{26}$$

In the above derivation of the system model it was assumed that there are no noise sources. In the next section, we model the noise in the system.

System Model with noise

First, we will add noise to the relative displacement with the assumption that it can be modeled by a random noise vector q_k such that the noise is Gaussian distribution with zero mean,$\hat{q_k}$ and covariance matrix,U_k .

$$q_k \sim N(\hat{q_k}, U_k) \tag{27}$$

where

$$\hat{q_k} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and

$$U_k = \mathrm{E}(q_k - \hat{q_k})(q_k - \hat{q_k})^T$$

$$U_k = \begin{bmatrix} \sigma^2_{q[\Delta S],k} & \sigma_{q[\Delta\theta],k}\sigma_{q[\Delta S],k} \\ \sigma_{q[\Delta\theta],k}\sigma_{q[\Delta S],k} & \sigma^2_{q[\Delta\theta],k} \end{bmatrix}$$

With the assumption that the noise sources are independent, the off-diagonal elements of the covariance matrix, U_k are equal to zero.The computation of variances $\sigma^2_{q[\Delta S],k}$ and $\sigma^2_{q[\theta],k}$ for the model is discussed in section.

The control input, or relative displacement can be expressed now as shown below.

$$u_k = j(d_k) + q_k \tag{28}$$

$$u_k = \begin{bmatrix} u_{[\Delta S],k} \\ u_{[\Delta\theta],k} \end{bmatrix} + \begin{bmatrix} q_{[\Delta S],k} \\ q_{[\Delta\theta],k} \end{bmatrix}$$

This makes u_k a random vector. Assuming, that $u_{[\Delta S],k}$ and $u_{[\theta],k}$ are deterministic, uncertainty in$u_k$equals the uncertainty in the noise term q_k.

The system noise can similarly be modeled by a random noise vector w_k such that the noise is Gaussian distribution with zero mean , $\hat{w_k}$ and covariance matrix, Q_k. This noise source is not directly related to the relative displacement but is inherent to the system.???

$$w_k \sim N(\hat{w_k}, Q_k) \tag{29}$$

where

$$\hat{w_k} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

and

$$Q_k = \mathrm{E}(w_k - \hat{w_k})(w_k - \hat{w_k})^T$$

$$Q_k = \begin{bmatrix} \sigma^2_{w[x],k} & \sigma_{w[y],k}\sigma_{w[x],k} & \sigma_{w[\theta],k}\sigma_{w[x],k} \\ \sigma_{w[x],k}\sigma_{w[y],k} & \sigma^2_{w[y],k} & \sigma_{w[\theta],k}\sigma_{w[y],k} \\ \sigma_{w[x],k}\sigma_{w[\theta],k} & \sigma_{w[\theta],k}\sigma_{w[x],k} & \sigma^2_{w[\theta],k} \end{bmatrix}$$

With the assumption that the noise sources are independent, the off-diagonal elements of the covariance matrix, Q_k are equal to zero.The computation of variances $\sigma^2_{w[x],k}$, $\sigma^2_{w[y],k}$ and $\sigma^2_{w[\theta],k}$ for the model is discussed in section.

The system can be expressed now as shown below.

$$x_k = f(x_{k-1}, u_{k-1}) + w_k \tag{30}$$

$$x_k = \begin{bmatrix} f_x(x_{k-1}, u_{k-1}) \\ f_y(x_{k-1}, u_{k-1}) \\ f_\theta(x_{k-1}, u_{k-1}) \end{bmatrix} + \begin{bmatrix} w_{[x],k-1}) \\ w_{[y],k-1}) \\ w_{[\theta],k-1}) \end{bmatrix} \tag{31}$$

w_k consists of noise sources that are not directly related to u_k. Now, x_k is a random vector and with every time step the system noise increases the variance. Thus, the variance of the location grows with every time step.

We have proposed the System Model so that we can implement Kalman Filter on it. Now, we need to prepare the Measurement Model for it.

Measurement Model with noise

Starting with the assumption that there is no noise in the measurement, z_k, it is simply a vector containing for each state variable a variable that takes on the value of the corresponding state variable. The measurement vector, z_k is

$$z_k = \begin{bmatrix} z_{[x],k} \\ z_{[y],k} \\ z_{[\theta],k} \end{bmatrix} \tag{32}$$

In our system, we use the motion capture system in the lab to localize the robot. The motion capture system acts as absolute sensor and provides us directly the position of the robot in x and y coordinates and its orientation. Thus, z_k in this case is simply expressed as 33

$$z_k = \begin{bmatrix} x_{[x],k} \\ x_{[y],k} \\ x_{[\theta],k} \end{bmatrix}$$

Now, we can add noise to the measurement model. We assume that the noise in the odometry can be modeled by a random noise vector v_k such that the noise is Gaussian distribution with zero mean,$\hat{v_k}$ and covariance matrix,R_k .

$$v_k \sim N(\hat{v_k}, R_k) \tag{33}$$

where

$$\hat{v_k} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

and

$$R_k = \mathrm{E}(v_k - \hat{v_k})(v_k - \hat{v_k})^T$$

$$R_k = \begin{bmatrix} \sigma^2_{v[x],k} & \sigma_{v[y],k}\sigma_{v[x],k} & \sigma_{v[\theta],k}\sigma_{v[x],k} \\ \sigma_{v[x],k}\sigma_{v[y],k} & \sigma^2_{v[y],k} & \sigma_{v[\theta],k}\sigma_{v[y],k} \\ \sigma_{v[x],k}\sigma_{v[\theta],k} & \sigma_{v[\theta],k}\sigma_{v[x],k} & \sigma^2_{v[\theta],k} \end{bmatrix}$$

With the assumption that the noise sources are independent, the off-diagonal elements of the covariance matrix, R_k are equal to zero.

The measurement model can be expressed now as shown below.

$$z_k = x_k + v_k \tag{34}$$

$$z_k = \begin{bmatrix} x_{[x],k} \\ x_{[y],k} \\ x_{[\theta],k} \end{bmatrix} + \begin{bmatrix} v_{[x],k-1}) \\ v_{[y],k-1}) \\ v_{[\theta],k-1}) \end{bmatrix}$$

Since the measurement noise v_k is a Gaussian vector, this makes z_k a random vector.

Kalman Filter

Kalman filtering is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone. Kalman filter is very popularly used in the localization of the robot.

Kalman Filtering is a recursive process involving 2 main components: prediction and innovation. Prediction is also known as 'Time update' and refers to update of state variables using previous state and control input. This step does not involve using measurement. Innovation which is also known as 'Correction' or 'Measurement update' refers to estimation of state variable using measurement.

Linear System Equations

Recall the linear system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{35}$$

where $w_k \sim N(0, Q_k)$ and $v_k \sim N(0, R_k)$.

We will proceed with the assumption that A, B, C, Q and R are constant for a given system.

Prediction or Time update:

$$\begin{aligned} \hat{x}_{k|k-1} &= A\hat{x}_{k-1|k-1} + Bu_{k-1} \\ \Sigma_{k|k-1} &= A\Sigma_{k-1|k-1}A^T + Q \end{aligned} \tag{36}$$

Innovation or Measurement update:

$$\begin{aligned} H_k &= \Sigma_{k|k-1}C^T\left(C\Sigma_{k|k-1}C^T + R\right)^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + H_k[y_k - C\hat{x}_{k|k-1}] \\ \Sigma_{k|k} &= [I - H_kC]\Sigma_{k|k-1} \end{aligned} \tag{37}$$

Non-Linear System Equations

Now consider a nonlinear extension to the Kalman filter. Now the system is

$$\begin{aligned} x_k &= f\left(x_{k-1}, u_{k-1}\right) + w_{k-1} \\ y_k &= h\left(x_k\right) + v_{k-1} \end{aligned} \tag{38}$$

where $w_k \sim N\left(0, Q_k\right)$ and $v_k \sim N\left(0, R_k\right)$.

We will proceed with the assumption that f(), h(), Q and R are constant for a given non linear system.

Prediction or Time update:

$$\begin{aligned} \hat{x}_{k|k-1} &= f\left(\hat{x}_{k-1|k-1}, u_{k-1}\right) \\ \Sigma_{k|k-1} &= J_f \Sigma_{k-1|k-1} J_f^T + Q \end{aligned} \tag{39}$$

Innovation or Measurement update:

$$\begin{aligned} H_k &= \Sigma_{k|k-1} J_h^T \left(J_h \Sigma_{k|k-1} J_h^T + R\right)^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + H_k[y_k - h\left(\hat{x}_{k|k-1}\right)] \\ \Sigma_{k|k} &= [I - H_k J_h]\Sigma_{k|k-1} \end{aligned} \tag{40}$$

where J_f and J_h are the jacobians.

J_f is the jacobian matrix with the partial derivatives of the system function f(· , ·) with respect to the state x, evaluated at the last state estimate $\hat{x}_{k-1|k-1}$ and control input u_{k-1}.

J_h is the jacobian matrix with partial derivatives of the measurement function h(.) with respect to the state x is evaluated at the prior state estimate $\hat{x}_{k|k-1}$.

Kalman Filter Equations for Odometry

$$\begin{bmatrix} x_{[x],k} \\ x_{[y],k} \\ x_{[\theta],k} \end{bmatrix} = \begin{bmatrix} f_x(x_{k-1}, u_{k-1}) \\ f_y(x_{k-1}, u_{k-1}) \\ f_\theta(x_{k-1}, u_{k-1}) \end{bmatrix} + \begin{bmatrix} w_{[x],k-1}) \\ w_{[y],k-1}) \\ w_{[\theta],k-1}) \end{bmatrix} \tag{41}$$

$$\begin{bmatrix} x_{[x],k} \\ x_{[y],k} \\ x_{[\theta],k} \end{bmatrix} = \begin{bmatrix} x_{[x],k-1} + u_{[\Delta]S,k-1} \cdot \cos(x_{[\Theta],k-1} - \frac{u_{[\Delta\Theta],k-1}}{2}) \\ x_{[y],k-1} + u_{[\Delta]S,k-1} \cdot \sin(x_{[\Theta],k-1} - \frac{u_{[\Delta\Theta],k-1}}{2}) \\ x_{[\theta],k-1} - u_{[\Delta\Theta],k-1}) \end{bmatrix} + \begin{bmatrix} w_{[x],k-1}) \\ w_{[y],k-1}) \\ w_{[\theta],k-1}) \end{bmatrix} \tag{42}$$

Prediction or Time update:

$$\begin{bmatrix} \hat{x}_{[x],k|k-1} \\ \hat{x}_{[y],k|k-1} \\ \hat{x}_{[\theta],k|k-1} \end{bmatrix} = \begin{bmatrix} \hat{x}_{[x],k-1|k-1} + u_{[\Delta]S,k-1} \cdot \cos(\hat{x}_{[\Theta],k-1|k-1} - \frac{u_{[\Delta\Theta],k-1}}{2}) \\ \hat{x}_{[y],k-1} + u_{[\Delta]S,k-1} \cdot \sin(\hat{x}_{[\Theta],k-1|k-1} - \frac{u_{[\Delta\Theta],k-1}}{2}) \\ \hat{x}_{[\theta],k-1|k-1} - u_{[\Delta\Theta],k-1}) \end{bmatrix} \tag{43}$$

$$\begin{aligned} \hat{x}_{k|k-1} &= f\left(\hat{x}_{k-1|k-1}, u_{k-1}\right) \\ \Sigma_{[x],k|k-1} &= J_f \Sigma_{[x],k-1|k-1} J_f^T + Q \end{aligned} \tag{44}$$

$$
\begin{aligned}
J_{f,k} &= \frac{\partial f(x)}{\partial x}\Big|_{x=\hat{x}_{k-1|k-1},u=u_{k-1}} \\
&= \begin{bmatrix} \frac{\partial f_x}{\partial x_{[x]}} & \frac{\partial f_x}{\partial x_{[y]}} & \frac{\partial f_x}{\partial x_{[\theta]}} \\ \frac{\partial f_y}{\partial x_{[x]}} & \frac{\partial f_y}{\partial x_{[y]}} & \frac{\partial f_y}{\partial x_{[\theta]}} \\ \frac{\partial f_\theta}{\partial x_{[x]}} & \frac{\partial f_\theta}{\partial x_{[y]}} & \frac{\partial f_\theta}{\partial x_{[\theta]}} \end{bmatrix}_{x=\hat{x}_{k-1|k-1},u=u_{k-1}} \\
&= \begin{bmatrix} 1 & 0 & -u_{[\Delta]S} \cdot \sin(x_{[\Theta]} - \frac{u_{[\Delta\Theta]}}{2}) \\ 0 & 1 & +u_{[\Delta]S} \cdot \cos(x_{[\Theta]} - \frac{u_{[\Delta\Theta]}}{2}) \\ 0 & 0 & 1 \end{bmatrix}_{x=\hat{x}_{k-1|k-1},u=u_{k-1}}
\end{aligned}
\tag{45}
$$

NO THIS SHUD BE 3X3
Next jacobian

$$
\begin{aligned}
J_{h,k} &= \frac{\partial h(x)}{\partial x}\Big|_{x=\hat{x}_{k|k-1}} \\
&= \begin{bmatrix} \frac{\partial h_x}{\partial x_{[x]}} & \frac{\partial h_x}{\partial x_{[y]}} & \frac{\partial h_x}{\partial x_{[\theta]}} \\ \frac{\partial h_y}{\partial x_{[x]}} & \frac{\partial h_y}{\partial x_{[y]}} & \frac{\partial h_y}{\partial x_{[\theta]}} \\ \frac{\partial h_\theta}{\partial x_{[x]}} & \frac{\partial h_\theta}{\partial x_{[y]}} & \frac{\partial h_\theta}{\partial x_{[\theta]}} \end{bmatrix}_{x=\hat{x}_{k-1|k-1},u=u_{k-1}} \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{x=\hat{x}_{k-1|k-1},u=u_{k-1}}
\end{aligned}
\tag{46}
$$

Conclusions